

Constraints (Default / Unique / Check)

Constraints are used to:

1. Define a default value for a column. This default value can be used when new rows are inserted (added) into a table.
 - Implemented via the definition of the default constraint.
2. Ensure that all values in a column or columns are unique.
 - Implemented via the definition of the unique constraint.
3. Define a domain of valid values that a column can assume.
 - Implemented via the definition of the check constraint.

Constraints can be defined:

1. When the table is initially created, as part of the Create Table statement or,
2. For an existing table, via the Alter Table statement (done in a later lesson).

We use the `Constraint` keyword to define a constraint. A single column can have multiple constraints, each with its own constraint keyword definition. Each constraint must have a unique name. The constraint name is used in SQL messages, so use descriptive constraint names. We will use a prefix, as part of the constraint name, to identify the type of constraint:

Prefix	Constraint Type
--------	-----------------

PK	Primary Key
FK	Foreign Key
CK	Check
DF	Default
UQ	Unique

A constraint can be defined as a column level constraint (applies to one column only) or sometimes as a table level constraint (applies to many columns in the table).

Note: constraint names in this document are shortened for readability and should not be considered appropriate names.

Default Constraint

The default constraint lets you define a value that is assigned to a column when the user does not specifically supply a value when a new row is added to the table (using the Insert statement). A default constraint can be defined on any column except:

- A column with the serial property (not discussed in this course)

The value of the default can be supplied via a constant (5.90) or a function (i.e. `Current_Date`). The default constraint name should use a prefix of `DF` (indicating this is a default constraint).

Syntax: [Constraint ***constraint_name***
Default {constant | function}]

Examples:

Default

Use the current date as the default for the DateReceived column

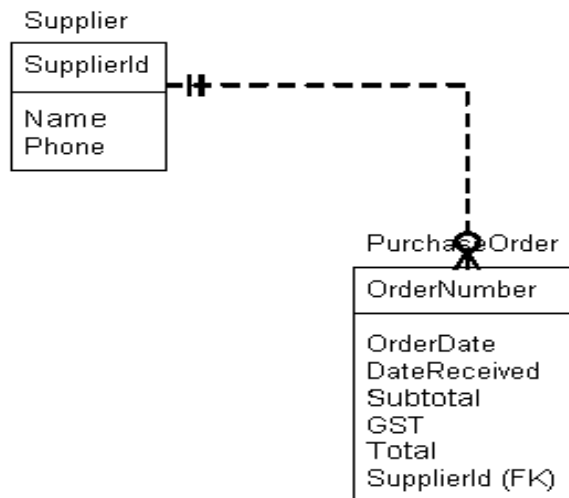
Constraint Definition

Constraint DF_DateReceived
Default Current Date

Use 5.90 as the default for the
HourlyRate column

Constraint DF_HourlyRate Default
5.90

A default constraint can only be defined at the column level.
Example:



To set a default of the current date for the OrderDate column in the PurchaseOrder table we use the Current_Date function to supply the system date as the default value.

Create Table PurchaseOrder

```

(
    OrderNumber    integer          not null
                                Constraint PK_PurchaseOrder Primary
Key,
    OrderDate      date             not null
                                Constraint DF_OrderDate
                                Default Current_Date,
    DateReceived   date             not null,
    SupplierID     integer          not null,
    SubTotal       decimal (6,2)    not null,
    GST            decimal (4,2)    not null,
  
```

); Total decimal (8,2) not null

Unique Constraint

The unique constraint makes sure that the values in one or more columns are not repeated. When applied to a single column, this ensures all entries in that column are different. When applied to multiple columns, this requires that the combination of values across those columns is unique, so no two rows can have exactly the same values.

A unique constraint does not consider null values to be equal.

Adding a unique constraint will automatically create a unique index on the column or group of columns used in the constraint.

Column Constraint Syntax

[Constraint ***constraint_name***] Unique

Table Constraint Syntax

[Constraint ***constraint_name***
Unique (***col_name*** [, ***col_name2*** [, . . . ***col_name32***])])

Example:

Students cannot use the same Email address, and multiple students cannot have the same first and last names.

Create Table Student

```
(
    StudentID      integer          not null,
    FirstName      varchar (30)      not null,
    LastName       varchar (30)      not null,
    Email          varchar (50)      not null
                                Constraint UQ_Student_Email
                                Unique,
    Constraint UQ_Student_Names Unique (FirstName, LastName)
);
```

Check Constraint

The Check constraint enables you to specify acceptable values, which can be inserted into a column in a table. As such, the check constraint lets you define a Domain for the column. The check constraint should be given a meaningful name and consists of an expression that evaluate to true or false. Whenever a new row of data is added to the table, or an existing column has its value changed, the new data will be evaluated using the expression in the check constraint. If the expression evaluates to true, the DBMS will add the new data to the table. If the expression evaluates to false, an error message is issued, and the new data is rejected.

Syntax:

```
Constraint constraint_name  
Check (expression)
```

The check constraint name should begin with the prefix CK (identifies the constraint as a check constraint) and be meaningful. Constraint names must be unique within the database. The syntax is identical whether the constraint is defined at the column level or the table level.

The expression in the check constraint:

- cannot contain a subquery
- must evaluate to true or false
- can be a compound Boolean expression
- can reference another column in the same table if the constraint is a table constraint

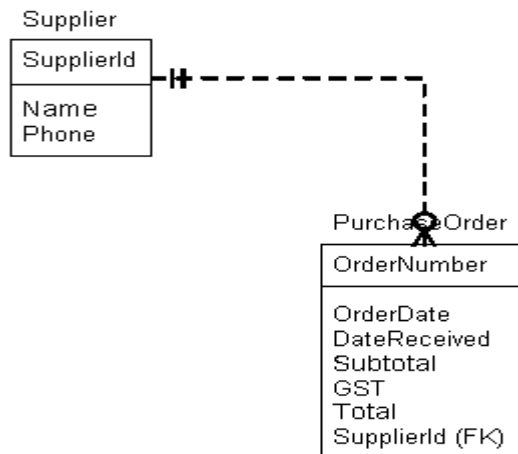
Examples of check constraints:

Column	Domain	Check Constraint
QuantitySold	Must be positive	Constraint CK_QuantitySold Check (QuantitySold > 0)
DateReceived	Must be >= the Date Ordered	Constraint CK_DateReceived Check (DateReceived >= DateOrdered)
CourseMark	Must be >= 0 and <= 100	Constraint CK_CourseMark Check (CourseMark Between 0 and 100) Or Constraint CK_CourseMark Check (CourseMark >= 0 and CourseMark <= 100)
PostalCode	Must follow the pattern A9A 9A9	Constraint CK_PostalCode Check (PostalCode ~ '[A-Z][0-9][A-Z] [0-9][A-Z][0-9]')

The expressions used in the check constraint are similar to expressions you have used in the Where clause.

If the expression of the check constraint refers to one column in a table, the constraint should be defined as part of the column definition. If the expression refers to two or more columns in the table, the constraint must be defined as a table level constraint.

Example:



Create Table Supplier

```

(
    SupplierID      integer          not null
                    Constraint PK_Supplier Primary Key,
    Name            varchar (100)      not null,
    Phone           char (14)          not null
                    Constraint CK_Phone
    Check (Phone ~ '\([0-9][0-9][0-9]\) [0-9][0-9][0-9]-[0-9][0-9][0-9]')
);
  
```

Note: Left and right brackets are “special” characters, so they need to be escaped with a slash

Create Table PurchaseOrder

```

(
    OrderNumber      integer          not null
                    Constraint PK_PurchaseOrder Primary
    Key,
    OrderDate        date             not null,
    DateReceived     date             not null,
    SupplierID       integer          not null
                    Constraint FK_PurchaseOrder_Supplier
                    References Supplier (SupplierID),
    SubTotal         decimal (6,2)     not null
                    Constraint CK_SubTotalMustBePositive Check
    (Subtotal > 0),
    GST              decimal (4,2)     not null
                    Constraint CK_GSTMustBePositive Check (GST > 0),
    Total            decimal (8,2)     not null
  )
  
```



```
        Constraint CK_TotalMustBePositive Check (Total >  
0),  
        Constraint CK_DateReceivedMustBeOnOrAfterOrderDate  
        Check (DateReceived >= OrderDate)  
);
```

Note that the check constraint for DateReceived and OrderDate is defined as a table level constraint (because it involves two columns) while all other check constraints are defined at the column level.

To test a check constraint:

1. Use pgAdmin to check the definition of the table. This ensures the constraint definition is in place and is correct.
2. Add a row to the table using data that violates the constraint. You should receive an error message, and the row should not be added to the table. This will be covered in a later lesson.